#4

Our Docket No: 42390.P9020

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of:	)		
Simon Knee, et al.	)	Examiner:	Unassigned
Application No: 09/823,616	)	Art Unit:	Unassigned
Filed: March 31, 2001	)		
For: FAST CLASSLESS INTER-DOMAIN ROUTING (CIDR) LOOKUPS	)		

Preliminary Amendment

Box Fee Amendment
Assistant Commissioner for Patents
Washington, D.C. 20231

Sir:

Prior to examination of the above-captioned case, the Applicant respectfully requests the Examiner to enter the following amendment and to consider the following remark:

FIRST CLASS CERTIFICATE OF MAILING

I hereby certify that I am causing the above-referenced correspondence to be deposited with the United States Postal Service as first class mail with sufficient postage on the date indicated below and that this paper or fee has been addressed to the Assistant Commissioner for Patents, Washington, D. C. 20231

June 20, 2001
Date of Deposit

Fran C. Rolfsen
Name of Person Mailing Correspondence

Fran C. Rolfsen
Signature

6-20-01
Date

AMENDMENTS -- CLEAN VERSION

In the Specification:

Please replace paragraph 0027 on page 10 with the following paragraph:

[0027] Generally speaking, the routing table 230 includes information that indicates on which port (the egress port) a particular address is located. In addition to the specific egress port through which to send the packet, the table might also store information about how the packet should be modified before forwarding. Such changes might include, but not be limited to, a new Media Access Control (MAC) destination address and Virtual LAN (VLAN) address. The routing table 230 typically includes entries having address data and a corresponding payload. The payload may be the forwarding instructions for the particular address or an indication (e.g., an index or a pointer) that can be used to find the forwarding instructions for the particular address. According to one embodiment, the routing table 230 is implemented in whole or part as a Content Addressable Memory (CAM), a random access memory (RAM), such as synchronous RAM (SRAM), or the like that may be implemented using hashing techniques. For example, a hash index generator (not shown), such as a cyclic redundancy checksum (CRC) generator may produce a hash index based on the destination IP address of the packet to be forwarded and the corresponding mask.

Please replace paragraph 0029 on page 11 with the following paragraph:

[0029] Briefly, in operation, the routing process 220 determines appropriate masks to apply to a search key associated with a received packet with reference to the mask table 240 and then queries the routing table 230 to identify the appropriate

forwarding instructions for the received packet. For example, the routing process 220 may retrieve an encoded mask vector corresponding to a destination network layer address contained in the received packet (e.g., a source or destination IP address) from the mask table 240 and then perform one or more address look-up requests using those of the masks indicated by the encoded mask vector to have a potential for matching an entry in the routing table 230.

Please replace paragraph 0033 on page 13 with the following paragraph:

[0033] At any rate, using this simplified example, a longest match search will be illustrated for a destination address 0B.01.02.F0 (the search key). In Internet Protocol (IP), there might be several entries of a routing table that match a particular address. In this example, network prefix 331 of route (entry) 330, network prefix 341 of route (entry) 340, and network prefix 351 of route (entry) 350 all match 0B.01.02.F0. However, to assure proper delivery of the packet, a network device must use the most specific matching entry, i.e., the entry having the longest mask. Importantly, to be considered a matching entry, the address associated with the entry must match a portion of the search key identified by its mask and the entry's mask length must be less than or equal to the search key's mask length.

Please replace paragraph 0051 on page 19 with the following paragraph:

[0051] At decision block 730, a determination is made whether any further masks are to be attempted. If MaskWord is not equal to zero, then at least one more mask remains to be tried and processing continues with processing block 740. However,

if MaskWord is equal to zero, then there are no further entries in the routing table 230 that can match the query at any length and longest match search processing is terminated. According to an alternative embodiment, the test for a zero MaskWord is eliminated and instead a special routing table entry is created to match a query of all zeroes and then terminate the search.

Please replace paragraph 0057 on page 21 with the following paragraphs:

[0057] The longest mask described by a MaskWord is found by forming each bit of the mask from the OR of the corresponding bit and all of the lower bits of the MaskWord.

```
Mask[0] = MaskWord[0];
Mask[1] = | MaskWord[1:0];
Mask[2] = | MaskWord[2:0];
Mask[3] = | MaskWord[3:0];
...
Mask[31] = | MaskWord[31:0];
```

where "| MaskWord[i,0]" is the REDUCTION OR operation which forms the OR of all bits in the identified range.

This approach of forming the mask employs some very large OR gates for the higher numbered bits. To avoid this people sometimes employ a ripple technique by observing that each bit of the mask already includes the OR of the corresponding bit and all of the lower bits of the MaskWord.

```
Mask[0] = MaskWord[0];
Mask[1] = MaskWord[1] || Mask[0];
Mask[2] = MaskWord[2] || Mask[1];
Mask[3] = MaskWord[3] || Mask[2];
...
Mask[31] = MaskWord[31] || Mask[30];
```

This ripple approach, sometimes called the recursive approach, has the advantage of reducing all of the logic to 31 two-input OR gates and the disadvantage of being slower because it takes 31 gate delays to determine the value of the highest order bit.

A look-ahead method compromises between these extremes of wide gates and long delays by using the ripple approach for most of the bits, but using an occasional wider gate to anticipate what the ripple value will be. Bits higher that the wider gates use their value rather than the ripple value.

```
Mask[0]  = MaskWord[0];
Mask[1]  = MaskWord[1] || Mask[0];
Mask[2]  = MaskWord[2] || Mask[1];
Mask[3]  = MaskWord[3] || Mask[2];
Mask[4]  = MaskWord[4] || Mask[3];
Mask[5]  = MaskWord[5] || Mask[4];

Mask[6]  = | MaskWord[6:0];  // anticipated value

Mask[7]  = MaskWord[7] || Mask[6];
Mask[8]  = MaskWord[8] || Mask[7];
Mask[9]  = MaskWord[9] || Mask[8];
Mask[10] = MaskWord[10] || Mask[9];
Mask[11] = MaskWord[11] || Mask[10];

Mask[12] = ( | MaskWord[12:7] ) || Mask[6]; // anticipated value

Mask[13] = MaskWord[13] || Mask[12];
Mask[14] = MaskWord[14] || Mask[13];
Mask[15] = MaskWord[15] || Mask[14];
...
Mask[31] = MaskWord[31] || Mask[30];
```

As mentioned earlier, those trying to implement this type of mask expansion in software or in FPGAs cannot efficiently use typical approaches available to ASIC designers. However, processors do have efficient instruction sets and many FPGAs have circuitry "tuned" to implement arithmetic functions. Therefore, if the target platform is

software running on a standard processor or an FPGA, the longest mask can be expanded from an encoded mask vector by the following equation, which provides a fast, compact solution:

$$\text{Mask} = (0 - \text{MaskWord}) \mid \text{MaskWord}$$

In the Claims:

Presented below are the claims, as amended, in a clean, unmarked format with changes entered and not marked. For the Examiner's convenience, all pending claims are presented herein. Claims that remain unchanged by this amendment are prefixed with "(Unchanged)."

Please amend claim 9 as follows:

1. (Unchanged) A method of performing a longest match search comprising:
receiving a search key;
determining a set of masks that when applied to the search key are known to have a potential for matching an entry in a routing table;
forming a routing table query based upon the search key and a longest mask of the set of masks; and
applying the routing table query to the routing table.

2. (Unchanged) The method of claim 1, further comprising:
removing the longest mask from the set of masks; and
continuing to apply additional routing table queries until either the set of masks is empty or a matching entry is found in the routing table.

3. (Unchanged) The method of claim 1, wherein the search key comprises an Internet Protocol (IP) address.

4. (Unchanged) The method of claim 3, wherein the IP address comprises a destination address.

5. (Unchanged) The method of claim 3, wherein the IP address comprises a source address.

6. (Unchanged) The method of claim 1, wherein said determining a set of masks comprises retrieving an encoded mask vector from a mask table based upon the search key, the encoded mask vector having N bits and capable of identifying N different length masks.

7. (Unchanged) The method of claim 1, wherein the longest mask of the set of masks is determined by the following equation: Mask = (0 – MaskWord) | MaskWord,

where:

MaskWord is an encoded mask vector, and

Mask is the longest mask identified by MaskWord.

8. (Unchanged) A packet forwarding device comprising:

a plurality of ports upon which packets are received and transmitted;

a routing processor coupled to the plurality of ports to determine an egress port of the plurality of ports for a packet received on an ingress port of the plurality of ports by performing a longest match search comprising one or more routing table queries;

a routing table, coupled to the routing processor, to provide the routing processor with a match indication and information regarding a matching routing table entry, if any, of a plurality of routing table entries stored therein in response to a routing table query; and

a mask table, coupled to the routing processor, to maintain encoded mask vectors identifying mask lengths of the plurality of routing table entries.

9. (Amended) The packet forwarding device of claim 8, wherein the encoded mask vectors comprise N-bits and are capable of representing N different masks.

10. (Unchanged) The packet forwarding device of claim 8, wherein the routing table comprises a Content Addressable Memory (CAM).

11. (Unchanged) The packet forwarding device of claim 8, wherein the one or more routing table queries are formed by applying a series of masks determined with reference to the mask table to a search key extracted from the received packet.

12. (Unchanged) A method of forwarding a packet comprising:

receiving a packet on an ingress port of a plurality of ports;

extracting a destination Internet Protocol (IP) address from a header of the packet;

using a portion of the destination IP address to index into a mask table to retrieve an encoded mask vector that identifies a series of masks to be applied to the destination IP address during a longest match search of a routing table, the series of masks representing those masks that are known to have a potential for matching an entry in the routing table when applied to the destination IP address;

identifying a longest matching entry in the routing table by performing the longest match search based upon the destination IP address and one or more of the series of masks; and

forwarding the packet to a network device associated with the destination IP address via an egress port of the plurality of ports identified by the longest matching entry.

13. (Unchanged) The method of claim 12, wherein the portion of the destination IP address comprises the most significant N bits of the destination IP address.

14. (Unchanged) The method of claim 12, wherein the encoded mask vector includes a plurality of mask length indicator bits that each indicate a mask length by virtue of their position within the encoded mask vector.

15. (Unchanged) The method of claim 12, further comprising updating the mask table to include a new encoded mask vector in response to receiving a new routing table entry.

16. (Unchanged) A machine-readable medium having stored thereon data representing sequences of instructions, the sequences of instructions which, when executed by a processor, cause the processor to:

receive a search key;

determine a set of masks that when applied to the search key are known to have a potential for matching an entry in a routing table;

form a routing table query based upon the search key and a longest mask of the set of masks; and

apply the routing table query to the routing table.

17. (Unchanged) The machine-readable medium of claim 16, wherein the longest mask of the set of masks is determined by the following equation: Mask = (0 – MaskWord) | MaskWord,

where:

MaskWord is an encoded mask vector, and

Mask is the longest mask identified by MaskWord.

18. (Unchanged) The machine-readable medium of claim 16, wherein the set of masks is determined by retrieving an encoded mask vector from a mask table based upon the search key, the encoded mask vector having N bits and capable of identifying N different length masks.

Please add the following claims:

19. (New) A method of forwarding a packet comprising the steps of:

a step for receiving a packet on an ingress port of a plurality of ports;

a step for extracting an Internet Protocol (IP) address from a header of the packet;

a step for using a portion of the IP address to index into a mask table to retrieve an encoded mask vector that identifies a series of masks to be applied to the IP address during a longest match search of a routing table, the series of masks representing those masks that are known to have a potential for matching an entry in the routing table when applied to the IP address;

a step for identifying a longest matching entry in the routing table by performing the longest match search based upon the IP address and one or more of the series of masks; and

a step for forwarding the packet to a network device based upon the longest matching entry.

20. (New) The method of claim 19, wherein the IP address comprises a destination IP address.

21. (New) The method of claim 19, wherein the IP address comprises a source IP address.

22. (New) The method of claim 19, wherein the encoded mask vector includes a plurality of mask length indicator bits that each indicate a mask length by virtue of their position within the encoded mask vector.

23. (New) A method of performing a longest match search comprising the steps of:

a step for receiving a search key;

a determination step for determining a set of masks that when applied to the search key are known to have a potential for matching an entry in a routing table;

a step for forming a routing table query based upon the search key and a longest mask of the set of masks; and

a step for applying the routing table query to the routing table.

24. (New) The method of claim 23, further comprising the steps of:

a step for removing the longest mask from the set of masks; and

a step for continuing to apply additional routing table queries until either the set of masks is empty or a matching entry is found in the routing table.

25. (New) The method of claim 23, wherein the search key comprises an Internet Protocol (IP) address.

26. (New) The method of claim 23, wherein said determination step comprises retrieving an encoded mask vector from a mask table based upon the search key, the encoded mask vector having N bits and capable of identifying N different length masks.

27. (New) The method of claim 1, wherein the longest mask of the set of masks is determined by the following equation: Mask = (0 – MaskWord) | MaskWord, where:

MaskWord comprises an encoded mask vector, and

Mask comprises the longest mask identified by MaskWord.

28. (New) The method of claim 7, further comprising:

isolating an endbit of the longest mask;

combining the longest mask with the inversion of the longest mask left-shifted one position; and

forming a subsequent routing table query based on the masked search key left-shifted one position and the endbit.

29. (New) A packet forwarding device comprising:

a plurality of interface means for receiving and transmitting packets;

routing processor means, coupled to the plurality of interface means, for determining an egress interface of the plurality of interface means for a packet received on an ingress interface of the plurality of interface means by performing a longest match search comprising one or more routing table queries;

a routing table means, coupled to the routing processor means, for providing the routing processor means with a match indication and information regarding a matching routing table entry, if any, of a plurality of routing table entries stored therein in response to a routing table query; and

a mask table means, coupled to the routing processor means, for maintaining encoded mask vectors identifying mask lengths of the plurality of routing table entries.

30. (New) The packet forwarding device of claim 29, wherein the encoded mask vectors comprise N-bits and are capable of representing N different masks.

31. (New) The packet forwarding device of claim 30, wherein the routing table means comprises a Content Addressable Memory (CAM).

32. (New) The packet forwarding device of claim 30, wherein the one or more routing table queries are formed by applying a series of masks determined with

reference to the mask table means to a search key extracted from the received packet.

REQUEST FOR APPROVAL TO AMEND DRAWING

Applicants propose correcting Figure 7 to change one symbol. Accordingly, Applicants have included herewith a red-inked original reflect~~ed~~ing the proposed change. No new matter has been added. The Examiner's approval is respectfully requested for the proposed change to Figure 7.

REMARK

Entry of the above-listed amendment is respectfully requested. It is respectfully submitted that no new matter has been introduced by this preliminary amendment.

INVITATION FOR A TELEPHONE INTERVIEW

The Examiner is invited to call the undersigned at 408-720-8598 if there remains any issue with the allowance of this case as amended.

CHARGE OUR DEPOSIT ACCOUNT

Please charge any shortage to our Deposit Account No. 02-2666.

Respectfully submitted,
BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP

Date: 6/20/01

Michael A. DeSanctis
Reg. No. 39,957

12400 Wilshire Boulevard
Seventh Floor
Los Angeles, CA 90025-1026
(303) 740-1980

Amendments -- Marked Up Version

In the Specification:

Presented below are the specification paragraphs, as amended, with changes marked. Insertions are underlined; deletions are bracketed.

Please replace paragraph 0027 on page 10 with the following paragraph:

[0027] Generally speaking, the routing table 230 includes information that indicates on which port (the egress port) a particular address is located. <u>In addition to the specific egress port through which to send the packet, the table might also store information about how the packet should be modified before forwarding. Such changes might include, but not be limited to, a new Media Access Control (MAC) destination address and Virtual LAN (VLAN) address.</u> The routing table 230 typically includes entries having address data and a corresponding payload. The payload may be the forwarding instructions for the particular address or an indication (e.g., an index or a pointer) that can be used to find the forwarding instructions for the particular address. According to one embodiment, the routing table 230 is implemented in whole or part as a Content Addressable Memory (CAM), a random access memory (RAM), such as synchronous RAM (SRAM), or the like that may be implemented using hashing techniques. For example, a hash index generator (not shown), such as a cyclic redundancy checksum (CRC) generator may produce a hash index based on the destination IP address of the packet to be forwarded and the corresponding mask.

Please replace paragraph 0029 on page 11 with the following paragraph:

[0029] Briefly, in operation, the routing process 220 determines appropriate masks to apply to a search key associated with a received packet with reference to the mask table 240 and then queries the routing table 230 to identify the appropriate forwarding instructions for the received packet. For example, the routing process 220 may retrieve an encoded mask vector corresponding to a destination network layer address contained in the received packet (e.g., a source or destination IP address) from the mask table 240 and then perform one or more address look-up requests using those of the masks indicated by the encoded mask vector to have a potential for matching an entry in the routing table 230.

Please replace paragraph 0033 on page 13 with the following paragraph:

[0033] At any rate, using this simplified example, a longest match search will be illustrated for a destination address 0B.01.02.F0 (the search key). In ~~Transmission Control Protocol/~~Internet Protocol (~~TCP/~~IP), there might be several entries of a routing table that match a particular address. In this example, network prefix 331 of route (entry) 330, network prefix 341 of route (entry) 340, and network prefix 351 of route (entry) 350 all match 0B.01.02.F0. However, to assure proper delivery of the packet, a network device must use the most specific matching entry, i.e., the entry having the longest mask. Importantly, to be considered a matching entry, the address associated with the entry must match a portion of the search key identified by its mask and the entry's mask length must be less than or equal to the search key's mask length.

Please replace paragraph 0051 on page 19 with the following paragraph:

[0051] At decision block 730, a determination is made whether any further masks are to be attempted. If MaskWord is ~~greater than~~not equal to zero, then at least one more mask remains to be tried and processing continues with processing block 740. However, if MaskWord is equal to zero, then there are no further entries in the routing table 230 that can match the query at any length and longest match search processing is terminated. According to an alternative embodiment, the test for a zero MaskWord is eliminated and instead a special routing table entry is created to match a query of all zeroes and then terminate the search.

Please replace paragraph 0057 on page 21 with the following paragraphs:

[0057] The longest mask described by a MaskWord is found by forming each bit of the mask from the OR of the corresponding bit and all of the lower bits of the MaskWord.

```
Mask[0] = MaskWord[0];
Mask[1] = | MaskWord[1:0];
Mask[2] = | MaskWord[2:0];
Mask[3] = | MaskWord[3:0];
...
Mask[31] = | MaskWord[31:0];
```

where "| MaskWord[i,0]" is the REDUCTION OR operation which forms the OR of all bits in the identified range.

This approach of forming the mask employs some very large OR gates for the higher numbered bits. To avoid this people sometimes employ a ripple technique by observing that each bit of the mask already includes the OR of the corresponding bit and all of the lower bits of the MaskWord.

```
Mask[0] = MaskWord[0];
Mask[1] = MaskWord[1] || Mask[0];
Mask[2] = MaskWord[2] || Mask[1];
Mask[3] = MaskWord[3] || Mask[2];
...
Mask[31] = MaskWord[31] || Mask[30];
```

This ripple approach, sometimes called the recursive approach, has the advantage of reducing all of the logic to 31 two-input OR gates and the disadvantage of being slower because it takes 31 gate delays to determine the value of the highest order bit.

A look-ahead method compromises between these extremes of wide gates and long delays by using the ripple approach for most of the bits, but using an occasional wider gate to anticipate what the ripple value will be. Bits higher that the wider gates use their value rather than the ripple value.

```
Mask[0]  = MaskWord[0];
Mask[1]  = MaskWord[1] || Mask[0];
Mask[2]  = MaskWord[2] || Mask[1];
Mask[3]  = MaskWord[3] || Mask[2];
Mask[4]  = MaskWord[4] || Mask[3];
Mask[5]  = MaskWord[5] || Mask[4];

Mask[6]  = | MaskWord[6:0];  // anticipated value

Mask[7]  = MaskWord[7] || Mask[6];
Mask[8]  = MaskWord[8] || Mask[7];
Mask[9]  = MaskWord[9] || Mask[8];
Mask[10] = MaskWord[10] || Mask[9];
Mask[11] = MaskWord[11] || Mask[10];

Mask[12] = ( | MaskWord[12:7] ) || Mask[6]; // anticipated value

Mask[13] = MaskWord[13] || Mask[12];
Mask[14] = MaskWord[14] || Mask[13];
Mask[15] = MaskWord[15] || Mask[14];
...
Mask[31] = MaskWord[31] || Mask[30];
```

As mentioned earlier, those trying to implement this type of mask expansion in software or in FPGAs cannot efficiently use typical approaches available to ASIC designers. However, processors do have efficient instruction sets and many FPGAs have circuitry "tuned" to implement arithmetic functions. Therefore, if the target platform is software running on a standard processor or an FPGA, the longest mask can be expanded from an encoded mask vector by the following equation, which provides a fast, compact solution:

$$\text{Mask} = ((\ 0 - \text{MaskWord}\ )\ |\ \text{MaskWord}$$

<u>In the Claims:</u>

Presented below are the claims, as amended, with changes marked. Insertions are underlined; deletions are bracketed.

<u>Please amend the claims as follows:</u>

1. (Unchanged) A method of performing a longest match search comprising:
   receiving a search key;
   determining a set of masks that when applied to the search key are known to have a potential for matching an entry in a routing table;
   forming a routing table query based upon the search key and a longest mask of the set of masks; and
   applying the routing table query to the routing table.

2. (Unchanged) The method of claim 1, further comprising:
removing the longest mask from the set of masks; and
continuing to apply additional routing table queries until either the set of masks is empty or a matching entry is found in the routing table.

3. (Unchanged) The method of claim 1, wherein the search key comprises an Internet Protocol (IP) address.

4. (Unchanged) The method of claim 3, wherein the IP address comprises a destination address.

5. (Unchanged) The method of claim 3, wherein the IP address comprises a source address.

6. (Unchanged) The method of claim 1, wherein said determining a set of masks comprises retrieving an encoded mask vector from a mask table based upon the search key, the encoded mask vector having N bits and capable of identifying N different length masks.

7. (Unchanged) The method of claim 1, wherein the longest mask of the set of masks is determined by the following equation: $\text{Mask} = (\ 0 - \text{MaskWord}\ ) \mid \text{MaskWord}$,
where:
MaskWord is an encoded mask vector, and
Mask is the longest mask identified by MaskWord.

8. (Unchanged) A packet forwarding device comprising:
a plurality of ports upon which packets are received and transmitted;
a routing processor coupled to the plurality of ports to determine an egress port of

the plurality of ports for a packet received on an ingress port of the plurality of ports by performing a longest match search comprising one or more routing table queries;

a routing table, coupled to the routing processor, to provide the routing processor with a match indication and information regarding a matching routing table entry, if any, of a plurality of routing table entries stored therein in response to a routing table query; and

a mask table, coupled to the routing processor, to maintain encoded mask vectors identifying mask lengths of the plurality of routing table entries.

9. (Amended) The packet forwarding device of claim 8, ~~the~~wherein encoded mask vectors comprise N-bits and are capable of representing N different masks.

10. (Unchanged) The packet forwarding device of claim 8, wherein the routing table comprises a Content Addressable Memory (CAM).

11. (Unchanged) The packet forwarding device of claim 8, wherein the one or more routing table queries are formed by applying a series of masks determined with reference to the mask table to a search key extracted from the received packet.

12. (Unchanged) A method of forwarding a packet comprising:

receiving a packet on an ingress port of a plurality of ports;

extracting a destination Internet Protocol (IP) address from a header of the packet;

using a portion of the destination IP address to index into a mask table to retrieve an encoded mask vector that identifies a series of masks to be applied to the destination IP address during a longest match search of a routing table, the series of masks representing those masks that are known to have a potential for matching an entry in the routing table when applied to the destination IP address;

identifying a longest matching entry in the routing table by performing the longest match search based upon the destination IP address and one or more of the series of masks; and

forwarding the packet to a network device associated with the destination IP address via an egress port of the plurality of ports identified by the longest matching entry.

13. (Unchanged) The method of claim 12, wherein the portion of the destination IP address comprises the most significant N bits of the destination IP address.

14. (Unchanged) The method of claim 12, wherein the encoded mask vector includes a plurality of mask length indicator bits that each indicate a mask length by virtue of their position within the encoded mask vector.

15. (Unchanged) The method of claim 12, further comprising updating the mask table to include a new encoded mask vector in response to receiving a new routing table entry.

16. (Unchanged) A machine-readable medium having stored thereon data representing sequences of instructions, the sequences of instructions which, when executed by a processor, cause the processor to:

receive a search key;

determine a set of masks that when applied to the search key are known to have a potential for matching an entry in a routing table;

form a routing table query based upon the search key and a longest mask of the set of masks; and

apply the routing table query to the routing table.

17. (Unchanged) The machine-readable medium of claim 16, wherein the longest mask of the set of masks is determined by the following equation: Mask = (0 – MaskWord) | MaskWord,

where:

MaskWord is an encoded mask vector, and

Mask is the longest mask identified by MaskWord.

18. (Unchanged) The machine-readable medium of claim 16, wherein the set of masks is determined by retrieving an encoded mask vector from a mask table based upon the search key, the encoded mask vector having N bits and capable of identifying N different length masks.

19. New

20. New

21. New

22. New

23. New

24. New

25. New

26. New

27. New

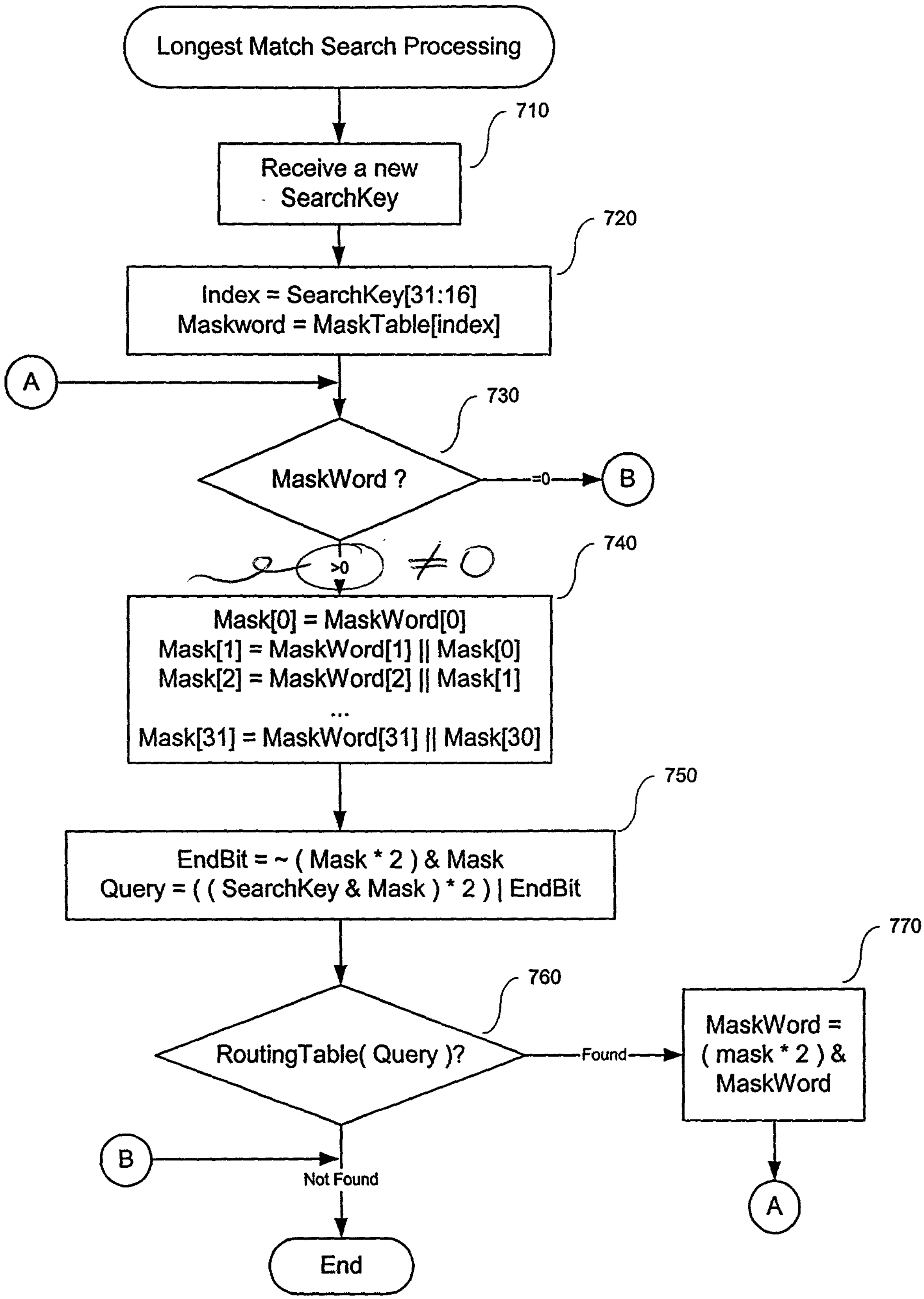
28. New

29. New

30. New

31. New

32. New



Longest Match Search Processing
710
Receive a new SearchKey
720
Index = SearchKey[31:16]
Maskword = MaskTable[index]
A
730
MaskWord ?
=0
B
>0
≠ 0
740
Mask[0] = MaskWord[0]
Mask[1] = MaskWord[1] || Mask[0]
Mask[2] = MaskWord[2] || Mask[1]
...
Mask[31] = MaskWord[31] || Mask[30]
750
EndBit = ~ (Mask * 2) & Mask
Query = ((SearchKey & Mask) * 2) | EndBit
760
RoutingTable(Query)?
Found
770
MaskWord = (mask * 2) & MaskWord
A
B
Not Found
End

Figure 7